Laboratorul 6

Lucrul în rețea

1 Retele de calculatoare

O retea de calculatoare presupune conectarea fizica a mai multor calculatoare numite si *noduri* (prin analogie cu grafurile, retelele pot fi modelate si analizate teoretic folosind teoria grafurilor) sau *host*-uri cu ajutorul unor medii de comunicare. Comunicarea efectiva dintre calculatoare folosind aceste medii de comunicare presupune doua premize fundamentale: calculatoarele implicate in comunicare trebuie sa se poate identifica unul pe altul si respectiv trebuie sa se poata "intelege" unul cu altul. Prima premiza este asigurata prin asignarea unei adrese de retea fiecarui calculator. Cea de-a doua este realizata cu ajutorul protocoalelor de comunicatie. In mod uzual, acestea determina si modul de adresare.

In cazul Internetului, protocoalele de comunicatie sunt denumite generic TCP/IP, desi in fapt e vorba de o suita de protocoale. Adresele asignate calculatoarelor se numesc adrese IP (Internet Protocol) si sunt folosite de protocoalele de comunicatie. O adresa de IP este compusa din 4 octeti separati de punct, de exemplu 192.168.0.1. Odata identificat, un calculator poate oferi o multitudine de servicii: web, mail, transfer de fisiere, etc. TCP/IP identifica aceste servicii prin porturi cu numere: 80 pentru web, 25 pentru mail, 21 pentru transfer de fisiere, samd. Ele se numesc *well-known ports* pentru ca sunt public cunoscute de toata lumea (ca intr-o carte de telefoane). O lista de well-known ports in sistemele Unix se poate gasi in fisierul `/etc/services`.

2 Domenii

Desi stau la baza comunicarii in internet, adresele de IP sunt mai rar folosite direct, calculatoarele avand in general un nume lizibil, e.g. `www.google.com`, care este asociat cu adresa lor de IP. Servere specializate numite `Domanin Name Servers (DNS)`, accesibile printr-un protocol de comunicatie special disponibil ca serviciu pe portul 53, raspund cererilor pe care alte calculatoare conectate la

internet, uzual definite ca fiind calculatoare client (sau pe scurt, clienti), le fac pentru a afla fie numele unui calculator data fiind adresa sa de IP, fie adresa de IP a unui calculator cunoscut dupa numele sau.

Concret, pentru a obține adresa de IP asociata unui host putem folosi mai multe metode. Comanda `nslookup(1)` (*name server look-up*) primește ca prim argument numele *host*-ului, asa-numitul Fully Qualified Domain Name sau FDQN, și, opțional, un al doilea argument care specifică ce server DNS să folosească pentru a căuta informația.

```
$ nslookup fmi.unibuc.ro
Server:         213.154.124.1
Address:        213.154.124.1#53

Non-authoritative answer:
Name:   fmi.unibuc.ro
Address: 193.226.51.6
```

În prima parte sunt afișate date legate de server-ul DNS folosit. A doua parte oferă informațiile cerute: numele și adresa. Dacă dorim să întrebăm un server anume (în exemplul de mai jos server-ul Google) îi punem adresa în al doilea argument:

```
$ nslookup fmi.unibuc.ro 8.8.8.8
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   fmi.unibuc.ro
Address: 193.226.51.6
```

Comanda `nslookup` poate fi lansata si in mod interactiv, caz in care ofera o mica linie de comanda care permite executia anumitor instructiuni, dupa cum se poate vedea mai jos:

```
$ nslookup
> server
Default server: 127.0.1.1
Address: 127.0.1.1#53
> server 1.1.1.1
Default server: 1.1.1.1
Address: 1.1.1.1#53
> set type=ptr
> 8.8.8.8
Server:         1.1.1.1
Address:        1.1.1.1#53

Non-authoritative answer:
8.8.8.8.in-addr.arpa    name = dns.google.
```

```
Authoritative answers can be found from:
> set type=a
> dns.google
Server:         1.1.1.1
Address:        1.1.1.1#53

Non-authoritative answer:
Name:   dns.google
Address: 8.8.4.4
Name:   dns.google
Address: 8.8.8.8
> set type=ns
> google.com
Server:         1.1.1.1
Address:        1.1.1.1#53

Non-authoritative answer:
google.com      nameserver = ns1.google.com.
google.com      nameserver = ns2.google.com.
google.com      nameserver = ns3.google.com.
google.com      nameserver = ns4.google.com.

Authoritative answers can be found from:
> set type=mx
> google.com
Server:         1.1.1.1
Address:        1.1.1.1#53

Non-authoritative answer:
google.com      mail exchanger = 10 smtp.google.com.

Authoritative answers can be found from:
> exit
```

Comanda *server* tipareste numele serverului la care apeleaza `nslookup` momentan pentru a rezolva cereri de DNS. Daca primeste ca parametru o adresa de IP sau un FDQN (Fully Qualified Domanin Name) a unui server DNS, va schimba adresa serverului DNS la care *nslookup* apeleaza. Comanda *set* in conjunctie cu parametrul *type* permite efectuarea de query-uri de FDQN (*type A*) care returneaza adresa IP corespunzatoare, query-uri de adrese IP (*type PTR*) care returneaza FDQN-ul corespunzator, query-uri pentru a afla serverul de DNS responsabil pt un anumit domeniu (*type NS*) sau pentru a afla serverul de mail (*type MX*) responsabil pentru un anumit domeniu.

Alte comenzi utile care funcționează similar sunt:

- `dig(1)`: `$ dig @8.8.8.8 fmi.unibuc.ro` – server-ul DNS trebuie prefi-

xat cu `@`

- `host(1)`: `$ host fmi.unibuc.ro 8.8.8.8` – server-ul DNS apare la sfarsitul comenzii
- `whois(1)`: `$ whois unibuc.ro` – informații despre domeniul principal, nu despre subdomeniul fmi)

Pentru a vedea dacă un *host* este accesibil in rețea putem folosi comanda `ping(1)`. Faceti distinctia intre conectare si accesibilitate. Un calculator poate fi fizic conectat la retea, dar temporar inaccesibil, fie din cauza unor defecte (hardware sau software) locale pe *host* fie din cauza unor defecte in reteaua din care face parte. De asemenea, se intampla adesea ca un *host* sa fie temporar inaccesibil pentru mentenanta. `ping` nu spune decat daca la momentul executiei comenzii *host*-ul este accesibil (se mai spune si *online*) sau nu.

```
$ ping fmi.unibuc.ro
PING fmi.unibuc.ro (193.226.51.6): 56 data bytes
64 bytes from 193.226.51.6: icmp_seq=0 ttl=50 time=7.317 ms
64 bytes from 193.226.51.6: icmp_seq=1 ttl=50 time=7.053 ms
64 bytes from 193.226.51.6: icmp_seq=2 ttl=50 time=6.925 ms
^C
--- fmi.unibuc.ro ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 6.925/7.098/7.317/0.163 ms
```

Observați că această comandă întâi găsește adresa *host*-ului și apoi comunică direct cu IP-ul acestuia. In Unix, dacă nu se specifică un număr de încercări, comanda va încerca până când utilizatorul o oprește cu `Ctrl-C`. În Windows, comanda încearcă de 4 ori implicit după care se oprește.

O alta comanda utila atunci cand vreti sa obtineti mai multe informatii despre accesibilitatea unui nod de retea din internet este `traceroute`. De pilda, daca vreti sa stiti peste cate *hop-uri* trece un pachet trimis de masina locala ca mai sus in comanda `nslookup` catre serverul DNS public al Google, puteti folosi comanda de mai jos:

```
$ traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  gateway (192.168.100.1)  0.837 ms  0.809 ms  7.898 ms
 2  10.0.33.143 (10.0.33.143)  5.829 ms  5.796 ms  5.880 ms
 3  10.12.100.1 (10.12.100.1)  5.839 ms  5.912 ms  5.871 ms
 4  10.220.187.54 (10.220.187.54)  5.905 ms  5.869 ms  5.931 ms
 5  10.220.208.244 (10.220.208.244)  19.839 ms  20.122 ms  45.225 ms
 6  72.14.216.212 (72.14.216.212)  25.055 ms 209.85.168.182 (209.85.168.182)
18.447 ms  16.041 ms
 7  * * 172.253.65.251 (172.253.65.251)  21.788 ms
 8  dns.google (8.8.8.8)  19.253 ms 142.251.65.217 (142.251.65.217)
22.249 ms 209.85.244.147 (209.85.244.147)  19.804 ms
```

Dupa cum observati, in cazul calculatorului de pe care a pornit comanda a fost nevoie sa se treaca de 7 routere pana sa se ajunga la destinatie (*dns.google*). Aceste routere intermediare se numesc in limbaj colocvial *hop-uri*.

Fișierul `/etc/hosts` este folosit pentru a defini manual perechi adresă IP – nume. In momentul in care se fac query-uri de DNS, clientul local DNS de pe calculator (asa numitul *DNS resolver*) cerceteaza fisierul `/etc/nsswitch.conf` pentru a detecta ordinea in care se cauta un server care sa rezolve cererea. In general, campul `hosts` al fisierului `/etc/nsswitch.conf` precizeaza ordinea de mai jos:

```
hosts:          files dns
```

Aceasta sintaxa ne spune ca in general fisierele locale (*files*), in cazul nostru `/etc/hosts`, sunt chestionate prioritar pentru rezolvarea numelui inaintea serverelor DNS. In cazul in care fisierul `/etc/hosts` contine rezolvarea dorita a numelui, DNS resolver-ul intoarce rezultatul gasit. Daca nu a gasit nicio intrare potrivita cautarii in `/etc/hosts`, resolver-ul va cauta sa acceseze un server DNS pentru a rezolva numele. Pe sistemele Unix, numele serverului DNS este uzual setat in fisierul `/etc/resolv.conf`.

Formatul `/etc/hosts` este simplu:

```
$ cat /etc/hosts
127.0.0.1       localhost
192.168.1.1     myserver
5.2.14.244      alex.unibuc.ro
```

Prima intrare din `/etc/hosts` ne indica asa-numita *adresa de loopback*. Aceasta este adresa locala a calculatorului si poate fi folosita ca orice adresa IP. Spre deosebire de o adresa IP din retea, adresa de loopback poate fi folosita oricand, chiar si atunci cand calculatorul nu este conectat fizic intr-o retea. In continuarea laboratorului veti folosi aceasta adresa de loopback pentru a exersa toate comenzile care urmeaza, atunci cand exercitiul nu indica expres o adresa de internet.

3 Servicii de retea

Atunci cand secventa de boot se incheie si kernelul da controlul procesului `init` (`systemd` in Linux), daca s-a selectat runlevel-ul care configureaza sistemul ca fiind conectat in retea (e.g., nivelurile 3 sau 5), am vazut la curs ca se executa o serie de scripturi de sistem de tip *rc* sau *run commands* care pornesc si monitorizeaza serviciile sistem din spatiul utilizator. In particular, pentru runlevel-urile care prevad functionarea in retea, aceste scripturi pornesc serviciile de retea.

Serviciile de retea in Linux se afla in general in directorul `/etc/init.d` si sunt manipulate cu ajutorul comenzii `service`. Aceasta foloseste numele serviciului (e.g., *ssh*) si comenzile pe care le intelege scriptul aferent serviciului. In general aceste comenzi sunt standardizate pentru toate scripturile lansate de `init`: {start, stop, restart, reload, etc}. Iata o secventa de pornire/oprire a serviciului de *ssh*:

```
$ service ssh start
$ service ssh stop
```

Aici `ssh` este un script cu acest nume din `/etc/init.d` care poate primi ca parametru string-urile {start, stop}, executand pe cale de consecinta pornirea, respectiv oprirea serviciului. O secventa echivalenta de comenzi este:

```
$ /etc/init.d/ssh start
$ /etc/init.d/ssh stop
```

Serviciile de retea (si in general toate serviciile sistem) pot fi activate respectiv dezactivate cu comanda `systemctl` care controleaza activitatea procesului `systemd` si a managerului de servicii. Activarea si respectiv dezactivarea unui serviciu nu presupune pornirea sau oprirea lui, ci doar marcheaza serviciul ca fiind legat de o procedura anume din sistem, de pilda bootarea calculatorului. Activarea/dezactivarea si pornirea/oprirea serviciilor sunt ortogonale: un serviciu poate activat fara a fi pornit, dupa cum poate fi pornit fara a fi activat. Ca un exemplu concret, urmatoarea comanda:

```
$ systemctl disable ssh
```

nu va avea nici un efect asupra serviciului de `ssh`. Daca era pornit, el va continua sa ruleze. In schimb, la bootarea calculatorului serviciul nu va mai fi pornit automat, va fi nevoie de apelul explicit al comenzii `service` pentru a il porni.

N.B. Executia comenzilor `service` si `systemctl` necesita drepturi de *root*.

4 Acces la distanță

4.1 Transfer de date prin `FTP`

Pentru a accesa un *host* ce servește date prin protocolul `FTP` se folosește cumanda `ftp(1)`.

```
$ ftp alex@fmi.unibuc.ro
$ ftp ftp://fmi.unibuc.ro
$ ftp fmi.unibuc.ro –P 2121
```

Multe servere oferă informații legate de acces și structura datelor pe server la momentul conectării. Este important de văzut dacă este oferit acces anonim, fără autentificare. În acest caz de obicei se folosește utilizatorul `anonymous` și, politicos, se trece ca parola adresa de email la care puteți fi contactați. Dacă nu doriți acest lucru, apăsați pur și simplu `Enter` când se cere parola.

O dată conectați va apărea promptul `ftp>` care indică faptul că vă aflați într-un `shell` specializat protocolului `FTP` . Comenzile de navigare și manipulare a fișierelor (dacă aveți dreptul) sunt aceleași ca cele învățate până acum în `shell`: `ls`, `cd`, `pwd`, `rmdir`, `chmod` etc. Pentru a vedea toate comenzile disponibile apelați la comanda `help`.

Pentru a urca sau coborî un fișier folosiți comenzile `put`, respectiv, `get`. Dacă aveți nevoie să efectuați operația pentru mai multe fișiere puteți folosi `mput` și `mget` (`m` de la *multiple*). Pentru a ieși folosiți `quit`.

Server-ele `FTP` sunt din ce în ce mai rare în spațiul public, dar comenzile și modul de lucru este comun cu înlocuitorii lor moderni (ex. `sftp`).

4.1.1 Instalarea si pornirea serviciului de FTP

Cel mai simplu mod de a exersa comanda `ftp` este sa porniti local serviciul corespunzator (cel mai probabil instalandu-l in prealabil):

```
$ apt install vsftpd
$ service vsftpd start
```

Obs: Daca ati instalat software-ul cu comanda `apt`, nu este necesara pornirea serviciului, procedura de instalare a serverului il si porneste automat. Pentru a verifica starea serviciului dupa instalare rulati comanda:

```
$ service vsftpd status
```

Odata ce serverul de FTP ruleaza, puteti folosi urmatoarea comanda:

```
$ ftp localhost
```

Puteti folosi la login contul *anonymous*? Daca nu, de ce? (v. pagina de manual).

4.2 Administrare prin `SSH`

4.2.1 Instalarea si pornirea serviciului de `SSH`

La fel ca mai sus, incepeti prin a porni serviciul de `SSH` (eventual instalandu-l in prealabil daca nu exista in sistem):

```
$ service ssh status
$ service ssh start     # daca serviciul e oprit
$ apt install openssh-server    # daca serviciul nu e instalat
```

4.2.2 Lucrul cu `SSH`

Pentru a executa anumite comenzi sau a configura servicii de pe un *host* aflat la distanta se folosește comanda `ssh(1)`.

```
$ ssh fmi.unibuc.ro
$ ssh alex@fmi.unibuc.ro
$ ssh alex@fmi.unibuc.ro -p 2222
```

Rezultatul acestei comenzi este deschiderea unui shell pe o masina aflata la distanta, identificata ca mai sus prin adresa de IP si/sau port. Numele de utilizator este fie implicit numele local al utilizatorului care lanseaza comanda `ssh`, fie cel precizat explicit in comanda inainte de caracterul @. Executia shell-ului la distanta esueaza daca utilizatorul nu reuseste sa se logheze in sistemul de la distanta cu parola de utilizator de pe sistemul respectiv.

O varianta mai comoda si mai sigura de autentificare, care nu presupune introducerea parolei de pe sistemul de la distanta, foloseste criptografia cu chei asimetrice. Aceasta metoda de autentificare presupune folosirea a doua chei: una *publica* si una *secreta/privata*. Cheia publica este cunoscuta tuturor (poate fi distribuita public) si poate fi preluata de sistemele de calcul care vor sa permita accesul pe baza ei. Cheia secreta este cunoscuta doar catre proprietarul contulului si trebuie pastrata in siguranta. Compromiterea ei impune automat generarea unei noi perechi de chei si inlocuirea celor vechi. Perechea de chei este folosita pentru autentificare si acces fara parola. Cheile sunt stocate de regula in directorul `~/.ssh/` din contul utilizatorului. Cheia publica, care foloseste uzual extensia `pub` este distribuita dupa generare pe sistemele de calcul in care se doreste accesul utilizatorului. Ea este adaugata pe sistemele respective intr-un fisier numit `~/.ssh/authorized_keys` care contine toate cheile publice ale utilizatorului care are dreptul sa utilizeze contul respectiv de pe masina aflata la distanta.

La lansarea comenzii `ssh` se verifica continutul directorului `~/.ssh/` si intâi se încearcă autentificarea prin chei asimetrice, daca acestea exista in director. Altfel, `ssh` recurge la procedura de fall-back si se incearca metoda clasică de autentificare cu utilizator și parolă.

O dată autentificați, suntem întâmpinați de un `shell` identic cu cel cu care am lucrat până acum doar ca ruleaza pe masina de la distanta, si ca atare toate comenzile sunt executate pe *host*-ul la distanță nu pe mașina proprie.

Pentru a executa o simplă comandă fără a mai intra în `shell`, putem specifica comanda imediat după *host*:

```
$ ssh fmi.unibuc.ro ls
```

Generarea unei chei asimetrice se face cu comanda `ssh-keygen(1)`.

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alex/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/alex/.ssh/id_rsa.
Your public key has been saved in /home/alex/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:ixhRJaJimfffiqTED8+ZGSp+ZMGtqwC/V7qsmxPTtAU alex@fmi
The key's randomart image is:
+---[RSA 2048]----+
|      . o..      |
|   o.Eo .        |
|.+o.o.           |
|...ooo.          |
|.   o++  S       |
|..++ooo...       |
|. +*o=....       |
|..o*@ * .        |
|.oOBoX .         |
```

```
+----[SHA256]-----+
```

Cheia publică are sufix `.pub` și se găsește în `/home/alex/.ssh/id_rsa.pub`. Cea privată se găsește în același loc dar fără sufix.

Implicit comanda `ssh-keygen(1)` generează chei `RSA`. Este recomandat să folosiți un algoritm mai nou cum ar fi `ed25519` sau `ecdsa`. Pentru aceasta folosiți argumentul `-t` *algoritm*:

```
$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
...
```

Cheile publice pentru cei care doriți să aibă acces pe contul dumneavoastră de pe un *host* (ex. calculatorul propriu, server web etc.) se pun în fișierul `.ssh/authorized_keys` din `$HOME`.

Pentru a adăuga o cheie publică `id_rsa.pub` folosiți

```
$ cat id_rsa.pub >> .ssh/authorized_keys
```

Pentru a transfera date prin `SSH` se folosește comanda `scp(1)` care se comportă aproape identic cu `cp(1)`. Diferența apare în specificarea sursei și destinației. Acestea sunt prefixate cu date legate de *host*.

```
$ scp hello.c fmi.unibuc.ro:
$ scp hello.c alex@fmi.unibuc.ro:code/
$ scp -r project/ alex@fmi.unibuc.ro:
```

Implicit, dacă nu este specificată nici o cale după `:`, transferul se face din/în directorul `$HOME` al utilizatorului. Dacă este specificată o cale, aceasta poate fi relativă `fmi.unibuc.ro:catalog` sau absolută `fmi.unibuc.ro:/etc/passwd`.

Adesea, comanda `scp` este folosita pentru a intia transferuri de date de dimensiuni mari, care pot dura foarte mult, facand impractica pastrarea deschisa a terminalului din care s-a lansat comanda. Pe de alta parte, inchiderea terminalului echivaleaza in mod uzual cu terminarea comenzii, ceea ce nu este de dorit. Exista programe care permit detasarea comenzii de terminalul de lucru, fapt ce permite inchiderea acestuia. Ulterior, cand utilizatorul reia sesiunea de lucru si deschide un nou terminal de lucru poate reatasa comanda noului terminal. In tot acest timp comanda a rulat in background si, presupunand ca nu au existat erori in executia ei, a progresat in realizarea obiectivului ei. Programe de acest tip care detaseaza o comanda de terminalul de lucru sunt de pilda `screen` sau `tmux`. Comanda `scp` atunci cand transfera cantitati mari de date se foloseste in mod uzual impreuna cu o comanda care permite detasarea de terminal. Mai jos aveti un exemplu de folosire a comenzii `screen` care detaseaza de terminal o comanda `ssh` care executa la distanta o comanda care dureaza mult, simulata in exemplul nostru de `sleep 300`:

```
$ screen ssh localhost sleep 300
```

Dupa autentificare (fara parola daca ati generat cu succes cheile asimetrice cf. procedurii de mai sus), comanda ssh va executa la distanta (in fapt local, pentru ca v-ati conectat la *localhost*) comanda `sleep 300`. Apoi puteti tasta

Ctrl-a-d pentru a detasa comanda `ssh` de terminal si veti reprimi controlul shell-ului (promptul). Puteti inchide acum terminalul. Deschideti un terminal nou si executati comanda urmatoare:

```
$ screen -ls
```

care va lista un identificator al comenzii detasate de vechiul terminal. Cu ajutorul comenzii `screen -r <identificator>` puteti reatasa comanda `ssh` detasata anterior la teminalul curent. Tastati Ctrl-c pentru a termina executia comenzii `sleep`.

O implementare similară `FTP` folosind protocolul `SSH` este `SFTP`. Pentru a accesa un server se folosește comanda `sftp(1)` în același mod în care folosim comanda `ssh(1)`. O dată autentificați, comenzile și modul de lucru sunt aproape identice cu cele din `FTP`. Excepție face faptul că modul anonim nu mai este disponibil.

5 Sarcini de laborator

1. Găsiți adresele IP pentru `google.com`, `fmi.unibuc.ro`, `wikipedia.org`. Adăugați câte o intrare pentru fiecare în `/etc/hosts`.

2. Inter-schimbați adresele de IP pentru `google.com` și `fmi.unibuc.ro`. Folosiți `ping(1)` pentru cele două *host*-uri înainte și după modificare. Apare vreo schimbare?

3. Scrieti un shell script care citeste tot continutul fisierului `/etc/hosts` si pentru fiecare linie de tip adresă IP – nume folositi comanda `nslookup` pentru a verifica adresa de IP a numelor din fisier. Daca `nslookup` va intoarce o alta adresa IP decat cea din `/etc/hosts` tipariti pe ecran mesajul "Bogus IP for <nume> in /etc/hosts !".
   Indicatie: O posibila solutie este sa folositi comenzile `cat` si `while` intr-un pipeline, `cat` pentru a afisa continutul `/etc/hosts` si `while` impreuna cu `read` pentru a itera prin continutul fisierului.

4. Inspectati cu un program de tip pager (`less/more`) continutul scriptului `/etc/init.d/ssh`. Intelegeti felul in care sunt folositi parametrii de apel `start|stop|status|reload`, etc.?

5. Accesați serverul FTP `ftp.vim.org` folosind un client `ftp` din linie de comanda. Navigați în directorul `pub/vim/pc/` și obțineți fișierul `vimXXsrc.zip` unde `XX` este cea mai recentă versiune pe care o găsiți în acel director (indiciu: folosiți `ls`).

6. Inspirati-va din ghidul `https://cloud.google.com/compute/docs/tutorials/basic-webserver-apache` pentru a vă crea pe masina locala o mașină virtuală Linux care servește pagini de Web. Porniti serviciul de `ssh` din masina virtuala, eventual instaland serviciul in prealabil daca nu exista pe masina virtuala. Folosiți `ssh-keygen(1)` pentru a genera o pereche

de chei publica-privata. Copiați cheia publică (cea cu extensia `.pub`) pe contul pe care il folositi pentru a accesa mașina virtuala si adaugati-o la continutul fisierului `authorized_keys` ca mai sus. Conectați-vă prin `ssh(1)` la noua mașină virtuală de pe calculatorul local.

7. Copiati un mic site Web de pe masina locala pe care o folositi pe serverul Web de pe masina virtuala creata anterior. Acest task revine la a copia directorul care contine site-ul Web pe mașina locala pe masina virtuala (server) folosind `scp(1)`. Noul director trebuie pus în `/var/www/html/student/`.
   Indicatii: Pentru Windows puteti transfera fisierele folosind comanda `scp` dintr-un shell `cygwin` sau cu ajutorul `putty`.
   Pentru a intra rapid in posesia unui mini site web, puteti downloada de pe internet cu comanda `wget` codul html al unui site existent. Folositi flag-urile -r (download recursiv) si -l pentru a limita adancimea arborelui de documente html descarcat. Ex: *wget -r -l 2 fmi.unibuc.ro*